What is claimed is:

1. A method for scheduling threads in a multi-processor computer system having an operating system and at least one cache, comprising the steps of:

storing in a first data structure thread ids for at least some of the threads associated with a context switch performed by the operating system, each of the thread ids uniquely identifying one of the threads;

storing in a second data structure a plurality of entries for a plurality of groups of contiguous cache lines, each of the plurality of entries arranged such that a thread id in the first data structure is capable of being associated with at least one of the contiguous cache lines in at least one of the plurality of groups of contiguous cache lines, the thread identified by the thread id having accessed the at least one of the contiguous cache lines in the at least one of the plurality of groups of contiguous cache lines;

mining for patterns in the plurality of entries in the second data structure to locate multiples of a same thread id that repeat with respect to at least two of the plurality of groups of contiguous cache lines; and

scheduling on a same processing unit the threads identified by the located multiples of the same thread id and any other threads identified by any other thread ids associated with the at least two of the plurality of groups of contiguous cache lines.

2. The method according to claim 1, further comprising the step of adding and removing a group to the plurality of groups of contiguous cache lines when a contiguous cache line in the group is accessed by a given thread and when all contiguous cache lines in the group are flushed, respectively.

3. The method according to claim 1, further comprising the step of restricting the plurality of groups to a finite number of groups.

4. The method according to claim 3, further comprising the step of determining when there exists the finite number of groups.

5. The method according to claim 3, wherein said mining step is performed when there exists the finite number of groups.

6. The method according to claim 1, wherein said mining step is performed upon a receipt of a command.

7. The method according to claim 1, wherein said mining step is performed at least one of continuously, at predefined intervals, and upon an occurrence of at least one predefined event.

8. The method according to claim 1, wherein said mining step is performed in at least one of software and hardware.

9. The method according to claim 1, wherein said second data structure is comprised of a plurality of rows and a plurality of columns.

10. The method according to claim 9, wherein each of the plurality of groups of contiguous cache lines corresponds to one of the plurality of rows.

11. The method according to claim 9, wherein each of the thread ids in the second data structure corresponds to one of the plurality of columns.

12. The method according to claim 11, wherein each of the plurality of groups of contiguous cache lines corresponds to one of the plurality of rows and the any other threads correspond to any of the plurality of columns that intersect any of the plurality of rows corresponding to the at least two of the plurality of groups.

13. The method according to claim 9, further comprising the step of allocating each of the plurality of rows to one of the plurality of groups of contiguous cache lines.

14. The method according to claim 10, further comprising the step of, for each of a cache line in a group in the plurality of groups of contiguous cache lines, storing an index of a row corresponding to the group containing the cache line in the cache line.

15. The method according to claim 1, wherein said method is implemented by a program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform said method steps.

16. A method for scheduling threads in a multi-processor computer system having an operating system and at least one cache, comprising the steps of:

storing in a first data structure thread ids for at least some of the threads associated with a context switch performed by the operating system, each of the thread ids uniquely identifying one of the threads;

storing in a second data structure a plurality of entries for a plurality of groups of contiguous cache lines, each of the plurality of entries arranged such that a thread id in the first data structure is capable of being associated with at least one of the contiguous cache lines in at least one of the plurality of groups of contiguous cache lines, the thread identified by the thread id having accessed the at least one of the contiguous cache lines in the at least one of the plurality of groups of contiguous cache lines;

mining for patterns in the plurality of entries in the second data structure to locate multiples of a same thread id that repeat with respect to at least two of the plurality of groups of contiguous cache lines; and

mapping the threads identified by the located multiples of the same thread id to at least one native thread.

17. The method according to claim 16, wherein the threads identified by the located multiples of the same thread comprise m threads and the at least one native thread comprises n threads, m and n being integers, m being greater than n.

18. The method according to claim 16, wherein said method further comprises the step of scheduling on a same processing unit the threads identified by the located multiples of the same thread id and any other threads identified by any other thread ids associated with the at least two of the plurality of groups of contiguous cache lines.

19. The method according to claim 16, further comprising the step of adding and removing a group to the plurality of groups of contiguous cache lines when a contiguous cache line in the group is accessed by a given thread and when all contiguous cache lines in the group are flushed, respectively.

20. The method according to claim 16, further comprising the step of restricting the plurality of groups to a finite number of groups.

21. The method according to claim 16, further comprising the step of determining when there exists the finite number of groups.

22. The method according to claim 16, wherein said mining step is performed when there exists the finite number of groups.

23. The method according to claim 16, wherein said mining step is performed upon a receipt of a command.

24. The method according to claim 16, wherein said mining step is performed at least one of continuously, at predefined intervals, and upon an occurrence of at least one predefined event.

25. The method according to claim 16, wherein said mining step is performed in at least one of software and hardware.

26. The method according to claim 16, wherein said second data structure is comprised of a plurality of rows and a plurality of columns.

27. The method according to claim 26, wherein each of the plurality of groups of contiguous cache lines corresponds to one of the plurality of rows.

28. The method according to claim 26, wherein each of the thread ids in the second data structure corresponds to one of the plurality of columns.

29. The method according to claim 28, wherein each of the plurality of groups of contiguous cache lines corresponds to one of the plurality of rows and the any other threads correspond to any of the plurality of columns that intersect any of the plurality of rows corresponding to the at least two of the plurality of groups.

30. The method according to claim 27, further comprising the step of allocating each of the plurality of rows to one of the plurality of groups of contiguous cache lines.

31. The method according to claim 27, further comprising the step of, for each of a cache line in a group in the plurality of groups of contiguous cache lines,

storing an index of a row corresponding to the group containing the cache line in the cache line.

32. The method according to claim 16, wherein said method is implemented by a program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform said method steps.

33. A method for scheduling threads in a multi-processor computer system having an operating system and at least one cache, comprising the steps of:

storing in a first data structure thread ids for at least some of the threads associated with a context switch performed by the operating system, each of the thread ids uniquely identifying one of the threads;

storing in a second data structure a plurality of entries for a plurality of groups of contiguous cache lines, each of the plurality of entries arranged such that a thread id in the first data structure is capable of being associated with at least one of the contiguous cache lines in at least one of the plurality of groups of contiguous cache lines, the thread identified by the thread id having accessed the at least one of the contiguous cache lines in

the at least one of the plurality of groups of contiguous cache lines;

identifying pools of threads in the plurality of entries in the second data structure such that each of the pools of threads comprises the threads identified by a same thread id that

forms a multiple with respect to one of the plurality of groups of contiguous cache lines, the multiple repeating with respect to at least two of the plurality of groups of contiguous cache lines; and

scheduling on a same processing unit the threads identified by the located multiples of the same thread id and any other threads identified by any other thread ids associated with the at least two of the plurality of groups of contiguous cache lines.